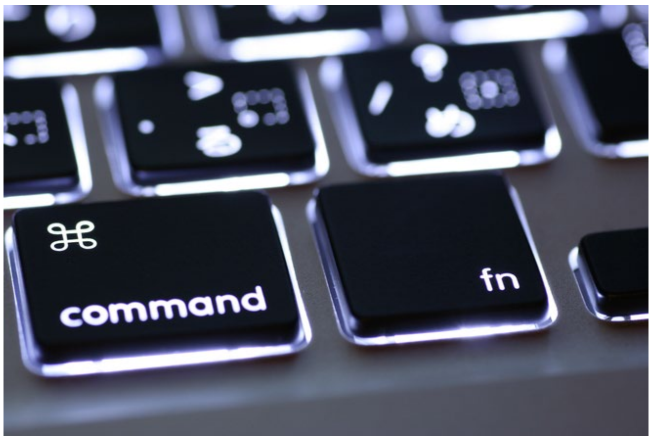
Structured PHP programming

Table of Contents

Functions 1
Creating a function 1
Calling a function 2
Defining parameters 2
Returning a value 2
Variable scope 3
Global variables 3
Include files 3
Difference Between include and require Statements 3
Example 4
Output buffering 14
Example 14

Functions



A function is a self-contained block of code that carries out a task. In your code you define a function and give it a name then call the function when the task needs to execute.

Functions are useful in programming for the following reasons:

- They avoid duplicating code – you write it once and call it whenever it is needed.
- They make it easier to eliminate errors – because functions are written once there is no need to copy and paste the same code block. This reduces the number of potential errors and makes code maintenance much easier.
- Functions can be reused in other projects.
- Functions help you break down the project into smaller parts. Helping to group and organise your code to make it easier to work with.

There are many built in functions in PHP like trim() which trims the leading and trailing white spaces in a string. You can also write your own functions.

Creating a function

A function is created by using the keyword “function”

Activity 1. Create a function

Create the following file:

```
<?php
   function greeting()
   {
      echo "Hello";
   }
?>
```

When you execute the code above nothing will happen. A function has been declared, the name of the function is “greeting”. The function has not been executed. To execute the function you need to call the function.

Calling a function

A function is called by using the name of the function followed by opening and closing brackets ().

Activity 2. Calling a function

Add the following line of code underneath the function declaration:

```
greeting();
```

Additional practice

1. Create a function that displays the current date and time and displays good morning if it is before 12 and good afternoon if it is after 12.

Defining parameters

Functions can accept values. To tell PHP your function accepts one or more values you need to create parameters. A parameter is a variable that holds the value passed to it when the function is called.

Activity 3. Adding parameters

Modify the "greeting" function to include a name:

```
<?php
   function greeting($firstName, $lastName)
   {
      echo "Hello " . $firstName . " " . $lastName;
   }

   greeting("John", "Smith");
?>
```

You can set a default value for a parameter by using the following syntax:

```
<?php
   function greeting($firstName, $lastName="Smith")
   {
      echo "Hello " . $firstName . " " . $lastName;
   }

   greeting("John");
?>
```

In the above example John Smith will be displayed. If the function call includes 2 parameters the second parameter will override the default value. For example calling:

```
Greeting("John", "Brown");
```

Will display John Brown.

Returning a value

Functions can return a value. For example the built in sqrt() function in PHP takes a number as input and returns the square root of that number.

Activity 4. Returning a value

This code returns the addition of 2 numbers:

```
<?php
   function add($number1, $number2)
   {
      $total = $number1 + $number2;

      return $total;
   }

   echo add(5,2);
?>
```

Additional practice

1. Create a function that returns the number of days in a month. It needs to take one input parameter, the month, and returns the number of days in the month.
2. Create a function that works out if a year is a leap year. It needs to take in one input parameter, the year, and returns true if the year is a leap year and false otherwise.
3. Combine the two to return 28 days for February if the year is not a leap year and 29 days if the year is a leap year.

Variable scope

Variables created inside a function, such as $total in the above example, are not available outside of the function. The variables created inside the function are said to be local. This is called the scope of a variable. So in the above example you cannot use the following statement:

```
function add($number1, $number2)
{
  $total = $number1 + $number2;

  return $total;
}

echo add(5,2);

echo $total;
```

This code will generate an error:

"Undefined variable"

Global variables

Sometimes you do want variables to be setup inside a function and be available outside of the function. You can do this by setting up the variable as a global variable. The syntax is:

```
global $myGlobal;
$myGlobal = "Hello there!";
```

The variable $myGlobal would be available from everywhere in the file.

Global variables can make it harder to debug your code when something goes wrong. You should try and avoid using global variables.

Include files

Including a file produces the same result as copying the script from the file specified and pasted into the location where it is called. With an include or a require statements you can include the contents of another file into your PHP code.

You can save a lot of time and work by using including files you can store a block of code in a separate file and include it wherever you want using the include() and require() statements instead of typing the entire block of code multiple times.

Difference Between include and require Statements

You can use either include or require. Typically, the require statement operates like include. The only difference is — the include statement will only generate a PHP warning but allow script execution to continue if the file to be included can't be found, whereas the require statement will generate a fatal error and stops the script execution.

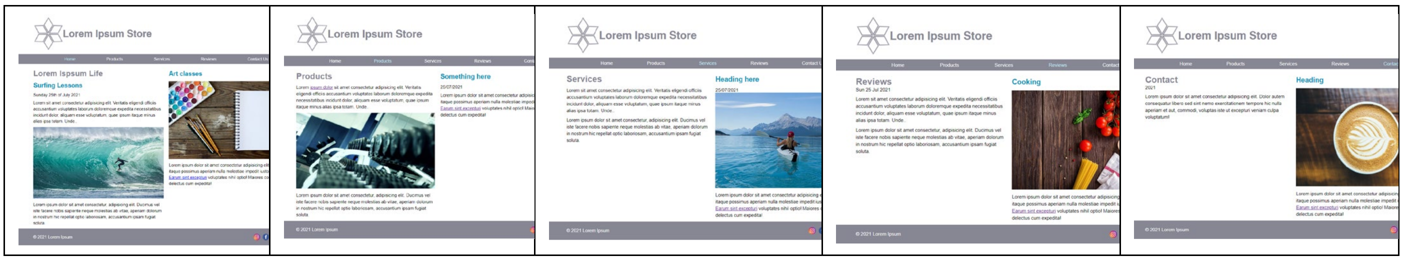
It is recommended to use the require statement if you're including the library files or files containing the functions and configuration variables that are essential for running your application, such as database configuration file.

You can also use include_once and require_once. The include_once and require_once statements include the file. This is a behaviour similar to the include and require statements, with the only difference being that if the code from a file has already been included, it will not be included again.

Example

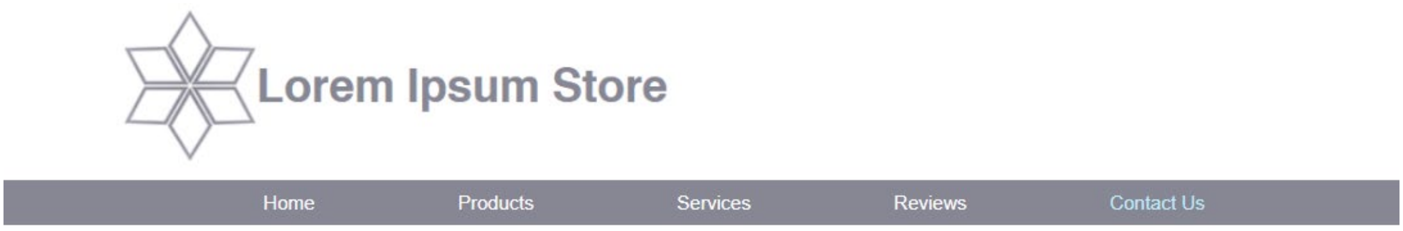
A typical example is including the header, footer and menu file within all the pages of a website.

If we have a website with 5 pages:



You can see for all 5 pages the header, navigation and footer is identical. We can create php files for each of the common portions.

Header:



Footer:

The include files would look something like:

Header: header.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title><?= $title ?></title>
    <link href="style.css" rel="stylesheet">

</head>
<body id="<?= $page ?>">
    <header class="wrapper">
        <a href="index.php">
            <img src="images/companyLogo.png" width="458" height="129" alt="Company Na
me">
        </a>
    </header>
    <nav>
        <ul class="wrapper">
            <li><a class="home" href="index.php">Home</a></li>
            <li><a class="products" href="products.php">Products</a></li>
            <li><a class="services" href="services.php">Services</a></li>
            <li><a class="reviews" href="reviews.php">Reviews</a></li>
            <li><a class="contact" href="contact.php">Contact Us</a></li>
        </ul>
    </nav>
```

Footer: footer.php

```
<footer>
        <div class="wrapper">
            <p> &copy; <?= date("Y"); ?>  Lorem Ipsum</p>
           <div class="socialMedia">
               <img src="images/instagram.png" width="25" height="25" alt="instagram">
               <img src="images/facebook.png" width="25" height="25" alt="facebook">
               <img src="images/paypal.png" width="25" height="25" alt="paypal">
               <img src="images/youtube.png" width="25" height="25" alt="youtube">
               <img src="images/twitter.png" width="25" height="25" alt="twitter">
           </div>
        </div>
    </footer>
</body>
</html>
```

The pages could then be created by including the above 2 files at the required location:

Home page:

```
<?php
$title = "Home - Lorem Ipsum";
$page = "home";

include "include/header.php";
?>

<main class="wrapper">
  <section id="column1">
    <h1>Lorem Ispsum Life</h1>
    <h2>Surfing Lessons</h2>
    <?php
    echo date("l jS \of F Y");
    ?>
    <p>Lorem sit amet consectetur adipisicing elit. Veritatis eligendi officiis accus
  antium voluptates laborum doloremque expedita necessitatibus incidunt dolor, aliquam
   esse voluptatum, quae ipsum itaque minus alias ipsa totam. Unde..</p>
    <img src="images/surf.jpg" width="500" height="273" alt="surfing">
    <p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. Ducimus vel iste fac
  ere nobis sapiente neque molestias ab vitae, aperiam dolorum in nostrum hic repellat
   optio laboriosam, accusantium ipsam fugiat soluta.
    </p>
  </section>
  <section id="column2">
    <h2>Art classes</h2>
    <img src="images/art.jpg" width="500" height="298" alt="art">
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Similique itaque poss
  imus aperiam nulla molestiae impedit iusto illum expedita.  voluptates nihil optio!
  Maiores commodi incidunt delectus cum expedita!</p>
  </section>
</main>

<?php
include "include/footer.php";
```

```
?>
```

Products page:

```
<?php
$title = "Products - Lorem Ipsum";
$page = "products";

include "include/header.php";
?>

<main class="wrapper">
  <section id="column1">
    <h1>Products</h1>
    <p>Lorem <a href="">ipsum dolor</a> sit amet consectetur adipisicing elit. Verita
  tis eligendi officiis accusantium voluptates laborum doloremque expedita necessitati
  bus incidunt dolor, aliquam esse voluptatum, quae ipsum itaque minus alias ipsa tota
  m. Unde..</p>
    <img src="images/weights.jpg" width="500" height="273" alt="weights">
    <p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. Ducimus vel iste fac
  ere nobis sapiente neque molestias ab vitae, aperiam dolorum in nostrum hic repellat
   optio laboriosam, accusantium ipsam fugiat soluta.
    </p>
  </section>
  <section id="column2">
    <h2>Something here</h2>
    <?php
    echo date("d/m/Y");
    ?>
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Similique itaque poss
  imus aperiam nulla molestiae impedit iusto illum expedita.  Earum sint excepturi
  voluptates nihil optio! Maiores commodi incidunt delectus cum expedita!
    </p>
  </section>
</main>

<?php
include "include/footer.php";
?>
```

Another approach is to place the header, navigation and footer in one file and holding a spot for the content that is different for each page:

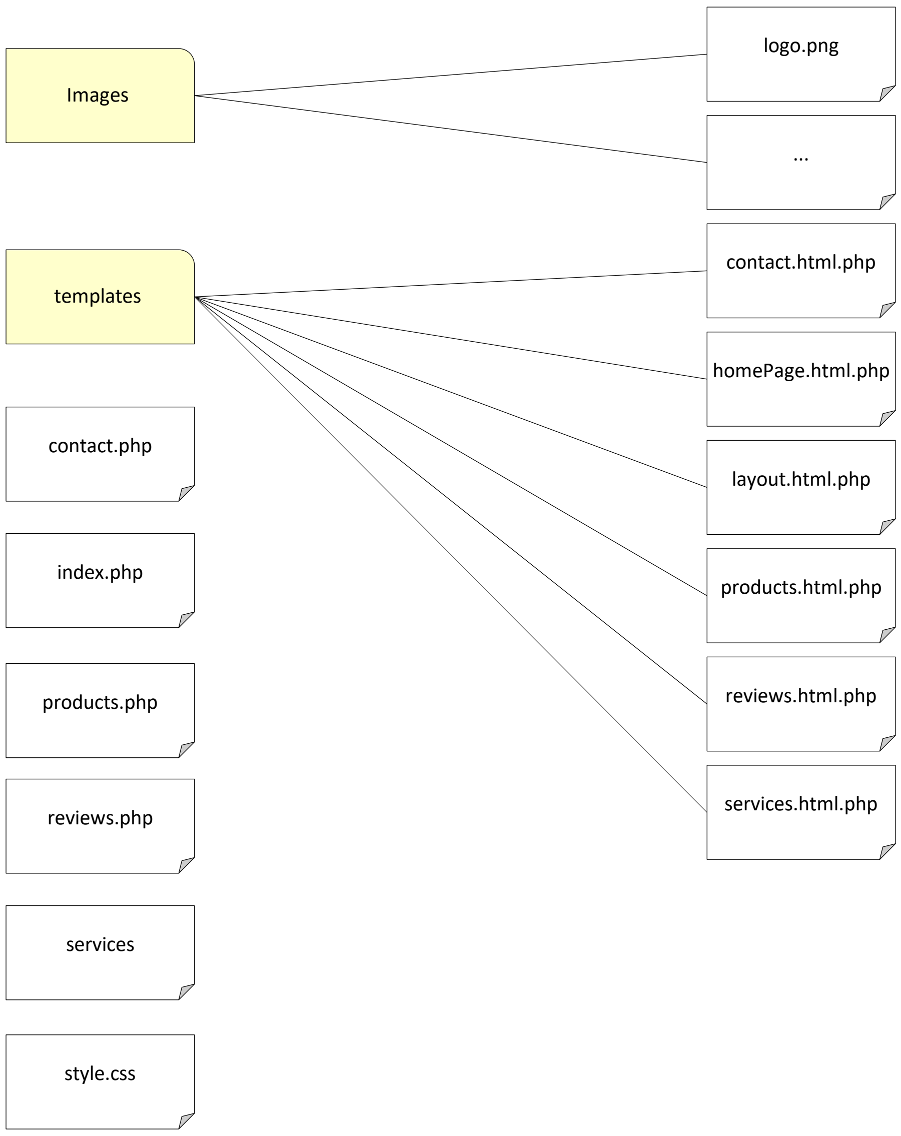
For this example, I have created a file called layout.html.php this file contains the common content for all the pages in the website.

Each individual page will include the layout.html.php page and any other HTML content. In addition to creating the template, sections of the pages being mainly HTML can also be stored inside other template files. This **helps separate HTML from the PHP code** and also **helps with code re-use**.

Eleven files will be created **some will have just PHP code inside them and some will have mainly HTML code**. To help identify the purpose of the files I have named the HTML files with a .html.php extension and the mainly PHP files with a .php extension:

File	Purpose
contact.html.php	Contains mainly HTML code for the contact page.
homePage.html.php	Contains mainly HTML code for the home page.
products.html.php	Contains mainly HTML code for the products page.
reviews.html.php	Contains mainly HTML code for the reviews page.
layout.html.php	This file contain the common HTML for all the pages in the website
services.html.php	Contains mainly HTML code for the services page.
contact.php	Contains the PHP code for the contact page and includes the contact.html.php and the layout.html.php content
index.php	Contains the PHP code for the home page and includes the homePage.html.php and the layout.html.php content
products.php	Contains the PHP code for the products page and includes the products.html.php and the layout.html.php content
reviews.php	Contains the PHP code for the reviews page and includes the reviews.html.php and the layout.html.php content
services.php	Contains the PHP code for the services page and includes the services.html.php and the layout.html.php content

When we create a website we usually separate the image files in another folder to make it easier to locate files. We can do the same for our template files (.html.php) files. I suggest placing these in a folder called template. So the site structure will look something like:



Note you never open the files in the templates folder, these are not full webpages.

templates/layout.html.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title><?= $title ?></title>
    <link href="style.css" rel="stylesheet">

</head>
<body id="<?= $page ?>">
    <header class="wrapper">
        <a href="index.php">
            <img src="images/companyLogo.png" width="458" height="129" alt="Company">
        </a>
    </header>
    <nav>
        <ul class="wrapper">
            <li><a class="home" href="index.php">Home</a></li>
            <li><a class="products" href="products.php">Products</a></li>
            <li><a class="services" href="services.php">Services</a></li>
            <li><a class="reviews" href="reviews.php">Reviews</a></li>
            <li><a class="contact" href="contact.php">Contact Us</a></li>
        </ul>
    </nav>
    <main class="wrapper">
        <?= $output ?>
    </main>
    <footer>
        <div class="wrapper">
            <p> &copy; <?= date("Y"); ?>  Lorem Ipsum</p>
           <div class="socialMedia">
               <img src="images/instagram.png" width="25" height="25" alt="instagram">
               <img src="images/facebook.png" width="25" height="25" alt="facebook">
               <img src="images/paypal.png" width="25" height="25" alt="paypal">
               <img src="images/youtube.png" width="25" height="25" alt="youtube">
               <img src="images/twitter.png" width="25" height="25" alt="twitter">
           </div>
        </div>
    </footer>
</body>
</html>
```

In the above file there are 4 sections that require php.

templates/contact.html.php

```
<section id="column1">
    <h1>Contact</h1>
    <?= $year ?>
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Dolor autem consequatu
r libero sed sint nemo exercitationem tempore hic nulla aperiam et aut, commodi, volup
tas iste ut excepturi veniam culpa voluptatum!</p>

</section>
```

```
<section id="column2">
    <h2>Heading </h2>
    <img src="images/coffee.jpg" width="490" height="339" alt="Coffee">
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Similique itaque possi
mus aperiam nulla molestiae impedit iusto illum expedita. <a href="">Earum sint except
uri</a> voluptates nihil optio! Maiores commodi incidunt delectus cum expedita!</p>
</section>
```

This page just contains mainly HTML with a bit of PHP. There are no doctypes, html, head, body etc as all these are inside the layout.html.php

templates/homePage.html.php

```
<section id="column1">
    <h1>Lorem Ispsum Life</h1>
    <h2>Surfing Lessons</h2>
    <?= $date ?>
    <p>Lorem sit amet consectetur adipisicing elit. Veritatis eligendi officiis accusa
ntium voluptates laborum doloremque expedita necessitatibus incidunt dolor, aliquam es
se voluptatum, quae ipsum itaque minus alias ipsa totam. Unde..</p>
    <img src="images/surf.jpg" width="500" height="273" alt="surfing">
    <p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. Ducimus vel iste face
re nobis sapiente neque molestias ab vitae, aperiam dolorum in nostrum hic repellat op
tio laboriosam, accusantium ipsam fugiat soluta.
    </p>
</section>
<section id="column2">
    <h2>Art classes</h2>
    <img src="images/art.jpg" width="500" height="298" alt="art">
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Similique itaque possi
mus aperiam nulla molestiae impedit iusto illum expedita. <a href="index.html">Earum s
int excepturi</a> voluptates nihil optio! Maiores commodi incidunt delectus cum expedi
ta!</p>
</section>
```

templates/products.html.php

```
<section id="column1">
    <h1>Products</h1>
    <p>Lorem <a href="">ipsum dolor</a> sit amet consectetur adipisicing elit. Veritat
is eligendi officiis accusantium voluptates laborum doloremque expedita necessitatibus
 incidunt dolor, aliquam esse voluptatum, quae ipsum itaque minus alias ipsa totam. Un
de..</p>
    <img src="images/weights.jpg" width="500" height="273" alt="weights">
    <p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. Ducimus vel iste face
re nobis sapiente neque molestias ab vitae, aperiam dolorum in nostrum hic repellat op
tio laboriosam, accusantium ipsam fugiat soluta.
    </p>
</section>
<section id="column2">
    <h2>Something here</h2>
    <?= $productDate ?>
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Similique itaque possi
mus aperiam nulla molestiae impedit iusto illum expedita. <a href="">Earum sint except
uri</a> voluptates nihil optio! Maiores commodi incidunt delectus cum expedita!</p>
```

```
</section>
```

templates/reviews.html.php

```
<section id="column1">
    <h1>Reviews</h1>
    <?php
    echo date("D d M Y");
    ?>
    <p>Lorem sit amet consectetur adipisicing elit. Veritatis eligendi officiis accusa
ntium voluptates laborum doloremque expedita necessitatibus incidunt dolor, aliquam es
se voluptatum, quae ipsum itaque minus alias ipsa totam. Unde..</p>

    <p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. Ducimus vel iste face
re nobis sapiente neque molestias ab vitae, aperiam dolorum in nostrum hic repellat op
tio laboriosam, accusantium ipsam fugiat soluta.
    </p>
</section>
<section id="column2">
    <h2>Cooking</h2>
    <img src="images/cooking.jpg" width="500" height="330" alt="Cooking">
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Similique itaque possi
mus aperiam nulla molestiae impedit iusto illum expedita. <a href="">Earum sint except
uri</a> voluptates nihil optio! Maiores commodi incidunt delectus cum expedita!</p>
</section>
```

templates/services.html.php

```
<section id="column1">
    <h1>Services</h1>
    <p>Lorem sit amet consectetur adipisicing elit. Veritatis eligendi officiis accusa
ntium voluptates laborum doloremque expedita necessitatibus incidunt dolor, aliquam es
se voluptatum, quae ipsum itaque minus alias ipsa totam. Unde..</p>

    <p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. Ducimus vel iste face
re nobis sapiente neque molestias ab vitae, aperiam dolorum in nostrum hic repellat op
tio laboriosam, accusantium ipsam fugiat soluta.
    </p>
</section>
<section id="column2">
    <h2>Heading here</h2>
    <?= $date ?>
    <img src="images/lake.jpg" width="500" height="333" alt="Lake">
    <p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Similique itaque possi
mus aperiam nulla molestiae impedit iusto illum expedita. <a href="">Earum sint except
uri</a> voluptates nihil optio! Maiores commodi incidunt delectus cum expedita!</p>
</section>
```

All of the above pages can be saved in the “templates” folder

The remaining pages can be located in the root folder of the website.

contact.php

```
<?php
    //set page specific values
    $title = "Contact - Lorem Ipsum";
    $page = "contact";

    //php contact for this page
    $year = date("Y");

    //start the buffer
    ob_start();

    //include template for this page
    include "templates/contact.html.php";

    //read the contents of the output buffer and store them
    //in $output variable so it can be used in layout.html.php

    $output = ob_get_clean();

    include "templates/layout.html.php";
?>
```

Again these pages do not contain the doctype, html, head, etc. elements. Notice ob_start() and ob_get_clean() these will be explained soon.

index.php

```
<?php
    //set page specific values
    $title = "Home - Lorem Ipsum";
    $page = "home";

    //php contact for this page
    $date = date("l jS \of F Y");

    //start the buffer
    ob_start();

    //include template for this page
    include "templates/homePage.html.php";

    //read the contents of the output buffer and store them
    //in $output variable so it can be used in layout.html.php

    $output = ob_get_clean();

    include "templates/layout.html.php";
?>
```

products.php

```
<?php
    //set page specific values
    $title = "Products - Lorem Ipsum";
    $page = "products";

    //php contact for this page
    $productDate = date("d/m/Y");

    //start the buffer
    ob_start();

    //include template for this page
    include "templates/products.html.php";

    //read the contents of the output buffer and store them
    //in $output variable so it can be used in layout.html.php

    $output = ob_get_clean();

    include "templates/layout.html.php";
?>
```

reviews.php

```
<?php
    //set page specific values
    $title = "Reviews - Lorem Ipsum";
    $page = "reviews";

    //php contact for this page
    $reviewsDate = date("D d M Y");

    //start the buffer
    ob_start();

    //include template for this page
    include "templates/reviews.html.php";

    //read the contents of the output buffer and store them
    //in $output variable so it can be used in layout.html.php

    $output = ob_get_clean();

    include "templates/layout.html.php";
?>
```

services.php

```
<?php
    //set page specific values
    $title = "Services - Lorem Ipsum";
    $page = "services";

    //php contact for this page
    $date = date("d/m/Y");

    //start the buffer
    ob_start();

    //include template for this page
    include "templates/services.html.php";
    //read the contents of the output buffer and store them
    //in $output variable so it can be used in layout.html.php

    $output = ob_get_clean();

    include "templates/layout.html.php";
?>
```

Output buffering

There's two types of operations you'll do in PHP: displaying content and trying to do data manipulation. Output buffering is a way to tell PHP to **hold some data before it is sent to the browser**. The content is stored in the buffer until you, the developer wants it displayed.

A few common output buffering functions:

- **ob_start()** turns on output buffering. It creates the buffer.
- **ob_get_contents()** gets all of the data stored in the buffer since the call to ob_start. Usually, this will be assigned to a variable.
- **ob_clean()** removes everything from the output buffer. Note that it does not output anything.
- **ob_flush()** outputs content from the buffer. Note that it does not erase the buffer.
- **ob_end_clean()** basically runs ob_get_contents(), erases the buffer, and turns off output buffering.
- **ob_end_flush()** outputs content from the buffer and ends output buffering. It does not erase the buffer.
- **ob_get_clean()** Get current buffer contents and delete current output buffer

A full list of functions is available in the php.net manual: https://secure.php.net/manual/en/ref.outcontrol.php

Example

```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="utf-8">
   <title>Output buffering</title>
</head>
<body>
   <h1>This is before the buffering</h1>
<?php
  // Turn on output buffering
  // There will be no output
  ob_start();

  $code = "ICT50615";
  $name = "Web diploma";
?>

<h2>Hi, the course code is <?= $code; ?>.</h2>
<p>and the name is <?= $name; ?>.</p>


<?php
  // Put all of the above ouptut into a variable
  $content = ob_get_contents();

  // Erase the buffer
  ob_end_clean();

  // All of the data that was in the buffer is now in $content
  //echo $content;
?>

<p>This is after the buffering</p>
</body>
</html>
```

By storing the content in the buffer we can rely on include files to separate the HTML content from the PHP code.